Preserving sparsity in dynamic network computation

Preserving Sparsity in Dynamic Network Computations

Francesca Arrigo and Desmond J. Higham

Abstract Time sliced networks describing human-human digital interactions are typically large and sparse. This is the case, for example, with pairwise connectivity describing social media, voice call or physical proximity, when measured over seconds, minutes or hours. However, if we wish to quantify and compare the overall time-dependent centrality of the network nodes, then we should account for the global flow of information through time. Because the time-dependent edge structure typically allows information to diffuse widely around the network, a natural summary of sparse but dynamic pairwise interactions will generally take the form of a large dense matrix. For this reason, computing nodal centralities for a time-dependent network can be extremely expensive in terms of both computation and storage; much more so than for a single, static network. In this work, we focus on the case of dynamic communicability, which leads to broadcast and receive centrality measures. We derive a new algorithm for computing time-dependent centrality that works with a sparsified version of the dynamic communicability matrix. In this way, the computation and storage requirements are reduced to those of a sparse, static network at each time point. The new algorithm is justified from first principles and then tested on a large scale data set. We find that even with very stringent sparsity requirements (retaining no more than ten times the number of nonzeros in the individual time slices), the algorithm accurately reproduces the list of highly central nodes given by the underlying full system. This allows us to capture centrality over time with a minimal level of storage and with a cost that scales only linearly with the number of time points.

Francesca Arrigo
University of Strathclyde, 16 Richmond St, Glasgow G1 1XQ, e-mail: francesca.arrigo@strath.ac.uk

Desmond J. Higham
University of Strathclyde, 16 Richmond St, Glasgow G1 1XQ, e-mail: d.j.higham@strath.ac.uk.

The work of the authors was supported by the Engineering and Physical Sciences Research Council under grant EP/M00158X/1.

1 Introduction

In network science, centrality measures assign to each node a value that summarises some aspect of its relative importance. Such measures arose in the social sciences, but have now become very widely used by researchers who wish to summarise important features of large, complex networks [5, 14, 19]. Because matrix representations of networks are typically sparse, and because centrality measures usually involve the solution of linear systems or eigenvalue problems, it is feasible to compute centrality measures on a current desktop computer for networks with, say, a number of nodes in the millions.

Our focus in this work is the case of time-dependent network sequences [8]. Such data sets may be regarded as three-dimensional tensors, where, along with the (i, j) coordinates that capture pairwise connectivity, we also have a third coordinate that represents time [1]. These types of connections arise, for example, when we record human-human digital interaction through social media, telecommunication or physical proximity. In [7] the concept of a *dynamic communicability matrix* was introduced, which converted the time sequence of networks into a single two-dimensional array, with (i, j) element summarising the ability of node i to communicate with node j, using the time-dependent sequence of edges recorded in the data. From this matrix, it is straightforward to compute centrality measures:

- *dynamic broadcast centrality* takes large values for nodes that are effective at distributing information,
- *dynamic receive centrality* takes large values for nodes that are effective at gathering information.

In a case study on Twitter data, this approach was seen to be successful, in the sense of correlating well with the independent views of social media experts [10]. It was also found to outperform the crude alternative of simply aggregating all edges into a single static network that forgets the time-ordering of the interactions; see [12] for further discussion. Tests in [4, 13] also showed that dynamic broadcast centrality can be effective at quantifying the potential for the spread of disease across time-ordered interactions.

However, as we explain in the next section, the computation of dynamic broadcast centrality can be expensive in terms of both storage and computation, as a result of inevitable matrix fill-in as temporal information accumulates. Our overall aim here is to address this issue by deriving a new algorithm that delivers good approximations to the original dynamic broadcast centrality measure while retaining the benefits of the sparsity present in the time slices.

We note that other approaches to computation of node centrality for time-dependent networks have been put forward. For example, [15, 16, 17] made use of paths rather than walks, which, for our purposes, leads to an infeasibly expensive algorithm. In [18] a block-matrix approach was suggested which allows centrality measures for static networks to be applied. However, as mentioned in [12], that formulation does not fully respect the arrow of time.

2 Background and Notation

In this section we recall some definitions and notation that will be used throughout. Let $t_0 < t_1 < \cdots < t_M$ be an ordered sequence of time points and let $\{\mathscr{G}^{[k]}\}_{k=0}^{M} = \{(\mathscr{V}^{[k]}, \mathscr{E}^{[k]})\}$ be a time-ordered sequence of unweighted graphs defined over n nodes. A graph is said to be unweighted when all its edges have the same weight, which can thus be assumed to be unitary. Consider the adjacency matrices $\{A^{[k]}\}_{k=0}^{M} = \{(a_{ij}^{[k]})\} \in \mathbb{R}^{n\times n}$ associated with these graphs at times $\{t_k\}_{k=0}^{M}$, whose entries are defined as

$$a_{ij}^{[k]} = \begin{cases} 1 \text{ if } (i,j) \in \mathscr{E}^{[k]} \\ 0 \text{ otherwise.} \end{cases}$$

In [7] the concept of a *dynamic walk of length* p was introduced to extend to the temporal case the well-known concept of a walk of length p in static networks. Loosely, we have a (possibly repeated) sequence of $p+1$ nodes connected by edges that appear in a suitable order. More precisely, a dynamic walk of length p from node i_1 to node i_{p+1} consists of a sequence of nodes $i_1, i_2, \ldots, i_{p+1}$ and a sequence of times $t_{r_1} \le t_{r_2} \le \cdots \le t_{r_p}$ such that $a_{i_m i_{m+1}}^{[r_m]} \neq 0$ for $m = 1, 2, \ldots, p$. We stress that more than one edge can share a time slot, and that time slots must be ordered but do not need to be consecutive.

The concept of dynamic walk was used to motivate the definition of the *dynamic communicability matrix*

$$Q^{[M]} = (I - \alpha A^{[0]})^{-1}(I - \alpha A^{[2]})^{-1} \cdots (I - \alpha A^{[M]})^{-1}, \tag{1a}$$

which can be defined equivalently via the iteration

$$Q^{[k]} = Q^{[k-1]}(I - \alpha A^{[k]})^{-1}, \quad k = 0, 1, \ldots, M, \tag{1b}$$

where $Q^{[-1]} = I$ is the identity matrix of order n, $0 < \alpha < 1/\rho^*$, and $\rho^* = \max_{k=0:M} \{\rho(A^{[k]})\}$ is the largest spectral radius among the spectral radii of the matrices $\{A^{[k]}\}$. Here the free parameter α plays the same role as in the classical Katz centrality measure for static networks [5, 9, 14]. For simplicity, our notation does not explicitly record the dependence of Q upon α.

To avoid overflow in the computations, a normalisation step $Q \mapsto Q/\|Q\|$ should follow each iteration in (1b). Throughout this work we use the Euclidean norm.

The requirement $\alpha < 1/\rho^*$ ensures that the resolvents in (1a) exist and can be expanded as $(I - \alpha A^{[k]})^{-1} = \sum_{p=0}^{\infty} (\alpha A^{[k]})^p$. It follows that the entries of $Q^{[k]}$ provide a weighted count of the dynamic walks between any two nodes in the networks using the ordered sequence of matrices $A^{[0]}, A^{[1]}, \ldots, A^{[k]}$, weighting walks of length p by a factor α^p. Hence, $(Q^{[k]})_{ij}$ is an overall measure of the ability of node i to send messages to node j.

Using the dynamic communicability matrix one can define and compare the broadcast and receive centrality of nodes by taking row and column sums of the matrix $Q^{[M]}$, respectively. The *broadcast centrality* of node i is defined as $b_i^{[M]} :=$

$\mathbf{e}_i^T Q^{[M]}\mathbf{1}$, where $\mathbf{e}_i \in \mathbb{R}^n$ is the ith column of I, the superscript "T" denotes transposition, and $\mathbf{1} \in \mathbb{R}^n$ is the vector of all ones. Similarly, the *receive centrality* of node j is defined as $r_j^{[M]} := \mathbf{1}^T Q^{[M]}\mathbf{e}_j$. It is straightforward to show that the latter satisfies a lower-dimensional, vector-valued iteration given by

$$\mathbf{r}^{[k]} := \mathbf{1}^T Q^{[k]} = \mathbf{r}^{[k-1]}(I - \alpha A^{[k]})^{-1}, \quad k = 0, 1, \ldots M,$$

with $\mathbf{r}^{[-1]} = \mathbf{1}$. The receive centrality of the nodes can thus be updated at each step by solving a single sparse linear system whose coefficient matrix is the latest network time slice. In particular, this means that we do not need to store and update the full matrix $Q^{[k]}$ to recover the receive centrality of nodes at level k. By contrast, to compute the broadcast centrality vector, $\mathbf{b}^{[M]} = Q^{[M]}\mathbf{1}$, we need access to the current dynamic communicability matrix at each step. Intuitively, this difference arises because,

- given a summary of how much information is flowing *into* each node, we can propagate this information forward when new edges emerge: receive centrality cares about where the information *terminates*, but
- a summary of how much information is flowing *out of* each node cannot be straightforwardly updated when new edges emerge: broadcast centrality cares about where the information *originates*.

Our focus here is on the natural setting where data is processed sequentially, with the centrality scores being updated as each new time slice $A^{[k]}$ arrives. As confirmed in Section 4 on a real data set, we then face a fundamental issue with the use of the dynamic communicability matrix: although the time slices are typically sparse, $Q^{[k]}$ generally evolves into a dense matrix. At this stage, computing dynamic communicability from (1b) requires us to store a full $O(n^2)$ matrix and solve at each subsequent time point a corresponding full linear system. In the next section, we therefore develop and justify an approximation where matrix fill-in is controlled so that the benefits of sparse matrix storage and computation are recovered.

3 Sparsification

To create a sparse approximation, $\widehat{Q}^{[k]}$, to the dynamic communicability matrix, $Q^{[k]}$, we first observe that the original iteration (1b) includes some traversals that are not very meaningful, e.g., repeated cycles $i \to j \to i \to j \to i \to j$ using the same undirected edge at the same time point. We thus use an "at most one edge per time point" alternative to (1b) so as to avoid considering these types of walks and similar ones:

$$\widehat{Q}^{[k]} = \widehat{Q}^{[k-1]}(I + \alpha A^{[k]}), \quad k = 0, 1, \ldots, M, \tag{2}$$

with $\widehat{Q}^{[-1]} = I$. As discussed in [7], this matrix product can be interpreted in terms of network combinatorics; at each time step a dynamic traversal can either wait, as described by the identity matrix I, or take a current edge, as described by latest adja-

cency matrix, $A^{[k]}$. In the latter case, the length of the walk (i.e., the number of edges used) has increased by one, and thus we multiply the corresponding matrix by α. An alternative interpretation is that we are using a second order Taylor approximation for each of the resolvents appearing in (1a). This simplification is likely to be reasonable when either (a) α is chosen to be small, so that short walks are favoured, or (b) the powers of $A^{[k]}$ do not grow rapidly with k (which is typically the case for sparse matrices).

As the time index k increases in (2) the number of nonzeros cannot decrease, and the matrix $\widehat{Q}^{[k]}$ will generally fill in. In order to produce a sparse approximation we will proceed iteratively. At each step we threshold the matrix at a level θ_k—this type of approach has been widely used in large scale machine learning, data mining, and signal processing; see, e.g., [2, 3] and references therein. Hence, for $k = 0, 1, \ldots, M$ we redefine the iteration to be

$$\widehat{Q}^{[k]} = \frac{\lfloor \widehat{Q}^{[k-1]}(I + \alpha A^{[k]})\rfloor_{\theta_k}}{\| \lfloor \widehat{Q}^{[k-1]}(I + \alpha A^{[k]})\rfloor_{\theta_k} \|_2}, \tag{3}$$

where $\widehat{Q}^{[-1]} = I$ and for any nonnegative matrix $C = (c_{ij})$, the matrix $\lfloor C \rfloor_{\theta_k}$ arises from setting to zero all entries where $c_{ij} \leq \theta_k$.

Remark 1. The matrices $\{\widehat{Q}^{[k]}\}_{k=0}^{M}$ are non-negative by construction.

3.1 A little twist

From a network science perspective, the approach just presented has a strong limitation. Imagine a user i of Twitter who remains inactive for a long time after each tweet. After such inactivity, the thresholding may zero out all entries in the ith row of one of the matrices $\widehat{Q}^{[k]}$. From that time, the ith row of the matrices appearing in (3) will always be zero, and no subsequent activity of node i will be registered by this approach.

To mitigate pathological behaviour of this type, we modify (3) so as to keep track at each step of the behaviour of those nodes corresponding to zero rows in the iteration matrix. Our final version of the iteration goes as follows:

$$\widehat{Q}^{[k]} = \lfloor \widehat{Q}^{[k-1]}(I + \alpha A^{[k]})\rfloor_{\theta_k} + m_k \mathscr{A}^{[k]}, \quad k = 0, 1, \ldots, M, \tag{4}$$

followed by normalisation, where $\widehat{Q}^{[-1]} = I$, m_k is the smallest nonzero entry of $\lfloor \widehat{Q}^{[k-1]}(I + \alpha A^{[k]})\rfloor_{\theta_k}$, $\mathscr{A}^{[k]} = \alpha W^{[k]} A^{[k]}$, and $W^{[k]} = \mathrm{diag}(w_1, w_2, \ldots, w_n) \in \mathbb{R}^{n \times n}$ is a diagonal matrix whose entries are

$$w_i = \begin{cases} 1 \text{ if } \mathbf{e}_i^T \lfloor \widehat{Q}^{[k-1]}(I + \alpha A^{[k]})\rfloor_{\theta_k} \mathbf{1} = 0 \\ 0 \text{ otherwise.} \end{cases}$$

The matrix $\mathscr{A}^{[k]}$ keeps track of those edges that appear at step k and would otherwise get lost. Indeed, the matrix product $W^{[k]}A^{[k]}$ returns a matrix that has nonzero entries (if any) only in the rows corresponding to those nodes that have either been inactive until step k or have broadcast very little information (which thus was thresholded in a previous iteration). The penalisation by α is added because we are taking one hop in the network. Finally, the multiplication by m_k comes from the fact that a poor choice of the parameter α may compromise the results. Indeed, the entries of $\mathscr{A}^{[k]}$ may be too large with respect to those appearing in $\lfloor \widehat{Q}^{[k-1]}(I+\alpha A^{[k]})\rfloor_{\theta_k}$, thus leading to a complete reshaping of the rankings. We refer the reader to Section 4 for an example of this issue.

Remark 2. It is possible for the contribution added by $m_k\mathscr{A}^{[k]}$ to be zero. This happens when the zero rows in $\lfloor \widehat{Q}^{[k-1]}(I+\alpha A^{[k]})\rfloor_{\theta_k}$ correspond to nodes that are not broadcasting information at step k.

Remark 3. Note that if $A^{[k]}=0$ for some k, then $\widehat{Q}^{[k]}=\widehat{Q}^{[k-1]}$, just as $Q^{[k]}=Q^{[k-1]}$.

3.2 On the thresholding parameters

The thresholding parameters $\{\theta_k\}$ are a key part of the sparsification process. Before explaining how we select these values in applications, we first describe the types of contributions that are removed from the approximation to the dynamic communicability matrix when the thresholding is performed. There are two key circumstances where the thresholding has an effect:

- the value of α^p dominates the contribution given by the products of the adjacency matrices, i.e., there are not too many walks of length p between the two nodes under consideration;
- the information has not moved from a certain node for a long time and the normalisation step has made the corresponding contribution smaller than the other entries.

In both cases, we are dismissing information that has little potential, as it is not diffused much. Clearly, an over-stringent selection of the parameters θ_k may lead to an excessive penalisation of these two types of behaviours. Our strategy is to make an initial choice for the maximum number of nonzeros that we will allow in the matrices $\widehat{Q}^{[k]}$, for $k=0,1,\ldots,M$. Then, as the iteration proceeds, the thresholding value θ_k is chosen so as to make $\lfloor \widehat{Q}^{[k-1]}(I+\alpha A^{[k]})\rfloor_{\theta_k}$ have approximately this desired level of sparsity.

We point out that the maximum number of nonzeros one wants to allow has to be at least $n+\mathtt{nnz}(A^{[0]})$, where $\mathtt{nnz}(A^{[0]})$ is the number of nonzeros in the matrix $A^{[0]}$. Consequently, $\theta_0<\alpha$. Indeed, if this is not the case, then we will have $\theta_k\geq\alpha$ for all k and therefore that $\widehat{Q}^{[k]}=I$ for all k.

3.3 Cost Comparison

We are now in a position to quantify, at least approximately, the computational benefits of using $\widehat{Q}^{[k]}$ in (4) rather than the exact matrix $Q^{[k]}$ in (1b) to compute dynamic broadcast communicability. Because the exact representation $Q^{[k]}$ becomes full in general, it follows that:

- We have reduced storage requirements by a factor of n.
- We have reduced the dominant computational task at each time step from solving n sparse linear systems to multiplying two sparse matrices. For general complex networks with no exploitable structure, if a standard iterative scheme is used to solve a sparse linear system, each matrix vector multiplication will cost $O(n)$ and thus the total cost to compute $Q^{[k]}$ by solving n such linear systems will be at least $O(n^2)$. Instead, the overall cost of computing the product of $\widehat{Q}^{[k-1]}$ times $A^{[k]}$ is $O(n)$, if we assume that there is a fixed number of active nodes at each time point. Thus, the cost has been reduced by a factor of n.

3.4 Comparing top K lists

The main goal of this work is to match the broadcast ranking of the nodes in an evolving network using a sparse approximation to the dynamic communicability matrix. As usual in network science, we are not interested in matching exactly the rankings of all nodes in the network, but rather to accurately capture the top $K \ll n$ most influential broadcasters. Although there is no perfect way to summarise and compare rankings, it is clear that generic correlation coefficients like Pearson's correlation coefficient or Kendall's tau have the major drawback in this context that they treat entire vectors, and hence all network nodes.

In order to compare the top K entries of two ranking vectors, an appropriate index is the *intersection similarity* [6]. This quantity is defined as follows: given two ranked lists x and y, consider the top K entries of each, which we denote x_K and y_K, respectively. Then, the top K intersection similarity between x and y is defined as

$$\mathrm{isim}_K(x,y) = \frac{1}{K}\sum_{i=1}^{K}\frac{|x_i \Delta y_i|}{2i}, \tag{5}$$

where Δ is the symmetric difference operator between two sets and $|S|$ denotes the cardinality of the set S. When the sequences contained in x and y are completely different, the intersection similarity between the two is maximum and equals 1. On the other hand, when $\mathrm{isim}_K(x,y) = 0$ for all K, then the two lists are identical.

It happens sometimes that the two lists differ in the *order*, but not in the *set of labels* of the nodes appearing in them. Behaviour of this type can be easily spotted by looking at the quantity

$$\ell_K(x,y) = \frac{|x_K \Delta y_K|}{2K}, \quad K = 2,3,\dots$$

If $\ell_K(x,y) = 0$ for some K we know that x_K and y_K are permutations of the same set of nodes.

4 Numerical tests

We have tested the new algorithm on large scale data sets involving email, voice call and on-line social interaction, and with various values of the parameter α. Due to space limitations we give representative results with the email data set Enron [11]. Here, a directed edge from node i to node j indicates that at least one message was sent from i to j in a one day period, including `to`, `cc`, and `bcc`. We have information over 1138 days starting 11 May 1999 for 151 Enron employees, Many of the adjacency matrices are empty, meaning that there are days during which no emails are sent. The largest spectral radius is $\rho^* = 4.17$, thus the upper limit for α is 0.24.

We allowed for a number of nonzeros proportional to $N = c\overline{n}$, where $\overline{n} = n + \frac{1}{M+1}\sum_{k=0}^{M} \mathtt{nnz}(A^{[k]})$ and $c = 10$. This is motivated by our aim to work only with matrices whose sparsity level is compatible with that of the individual network time slices. Further testing has shown that the performance is not sensitive to c.

4.1 Adaptive Scaling

Before testing the performance of (4), in this subsection we discuss the effect of including the multiplication by m_k. In Section 3 we argue that setting $m_k \equiv 1$ for all $k = 0,1,\dots,M$ in (4) may lead to poor results. Clearly, this is not always the case, but, as we will see here, this choice together with a compounding choice of the downweighting parameter α, may result in a complete misplacement of the top ranked broadcasters in the network.

We compute the broadcast centrality vector $Q^{[M]}\mathbf{1}$ and our approximation vector $\widehat{Q}^{[M]}\mathbf{1}$ for seven different values of the downweighting parameter:

$$\alpha = \frac{0.01}{\rho^*}, \frac{0.1}{\rho^*}, \frac{0.25}{\rho^*}, \frac{0.5}{\rho^*}, \frac{0.75}{\rho^*}, \frac{0.85}{\rho^*}, \frac{0.9}{\rho^*}.$$

Figure 1 displays the evolution of the intersection similarity between the top $K = 1,2,\dots,20$ entries of the vectors $Q^{[M]}\mathbf{1}$ and $\widehat{Q}^{[M]}\mathbf{1}$ versus K for the different values of α. The left plot contains the results when $m_k \equiv 1$, while the right plot contains the results when m_k is adapted by setting it to be equal to the smallest nonzero entry of the matrix $\lfloor \widehat{Q}^{[k-1]}(I + \alpha A^{[k]}) \rfloor_{\theta_k}$ at each iteration.

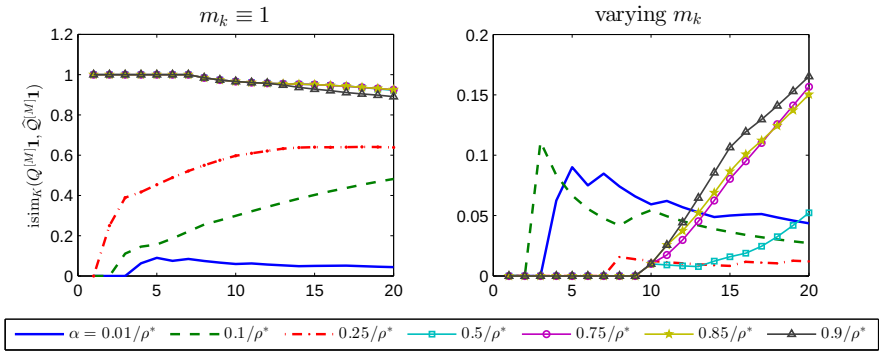

Fig. 1 Evolution of the intersection similarity $\text{isim}_K(Q^{[M]}\mathbf{1}, \widehat{Q}^{[M]}\mathbf{1})$ versus K, for different choices of the downweighting parameter α. Left: $m_k \equiv 1$. Right: m_k is set at each iteration as the smallest nonzero entry of $\lfloor \widehat{Q}^{[k-1]}(I + \alpha A^{[k]}) \rfloor_{\theta_k}$. Note the difference in vertical axis range.

These results show that when $m_k \equiv 1$ the intersection similarity between the two vectors can be maximum even when comparing only a few top ranked nodes for α as small as $0.5/\rho^*$. The right hand plot in the figure shows how an adaptive choice of m_k can work successfully over a wide range of α choices.

4.2 Centrality Approximation

We now assess the effectiveness of iteration (4) at approximating the broadcast centrality rankings. Using $\alpha = 0.01$, the number of nonzero entries in the dynamic communicability matrix is $\texttt{nnz}(Q^{[M]}) = 21097$. Note that $n^2 = 22801$, so the matrix is 92.5% full. Figure 2 scatter plots the resulting approximation to the broadcast and receive centrality vectors against $Q^{[M]}\mathbf{1}$ and $\mathbf{1}^T Q^{[M]}$, respectively. We observe a good linear correlation at the high end for both cases, indicating that our method correctly identifies important nodes. The number of nonzeros in the final approximation matrix $\widehat{Q}^{[M]}$ is $= 1676$, so the level of sparsity has been reduced to around 7.4%.

In Table 1 we list the top 10 ranked nodes according to the broadcast centrality. The first row contains the true result, obtained by ranking the nodes according to $Q^{[M]}\mathbf{1}$; in the second row we list the top 10 broadcasters according to the ranking derived from $\widehat{Q}^{[M]}\mathbf{1}$ and, finally, the last row displays the result obtained when the nodes are ranked according to their aggregate out-degree: $\sum_{k=0}^{M} A^{[k]}\mathbf{1}$. As $\alpha \to 0$, the ranking obtained using the dynamic communicability matrix approaches that obtained using the aggregate out-degree; see, e.g., [4, 7]. Clearly, however, $\alpha = 0.01$ is not close enough to zero for this effect to be observed.

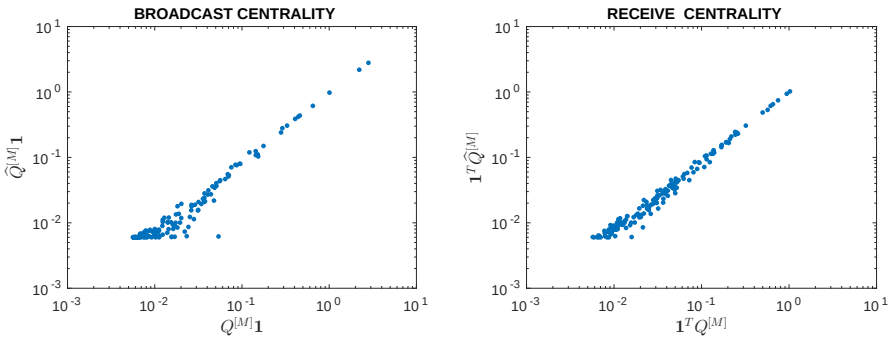

Fig. 2 Comparison of exact (horizontal) and approximate (vertical) centralities.

Table 1 Top 10 ranked nodes: exact, approximate and with aggregate out-degree.

$Q^{[M]}\mathbf{1}$	48	67	147	73	13	50	137	49	9	139
$\widehat{Q}^{[M]}\mathbf{1}$	48	67	147	73	13	50	137	49	9	139
out-degree	67	50	141	13	48	69	107	147	73	70

Tables 2-3 contain the values of $\text{isim}_K(Q^{[M]}\mathbf{1},\widehat{Q}^{[M]}\mathbf{1})$ for $K = 1,2,\ldots,20$ and $\ell_K(Q^{[M]}\mathbf{1},\widehat{Q}^{[M]}\mathbf{1})$ for $K = 2,3,\ldots,20$. We see that the new method correctly orders the top 11 broadcasters in the network and correctly identifies the top 20.

Table 2 Intersection similarity between the top $K = 1,2,\ldots,20$ ranked nodes in $Q^{[M]}\mathbf{1}$ and $\widehat{Q}^{[M]}\mathbf{1}$.

K	1	2	3	4	5	6	7	8	9	10
isim_K	0	0	0	0	0	0	0	0	0	0
K	11	12	13	14	15	16	17	18	19	20
isim_K	0	0.01	0.02	0.03	0.03	0.03	0.03	0.03	0.03	0.03

Table 3 Evolution of $\ell_K(Q^{[M]}\mathbf{1},\widehat{Q}^{[M]}\mathbf{1})$ for $K = 2,3,\ldots,20$.

K		2	3	4	5	6	7	8	9	10
ℓ_K		0	0	0	0	0	0	0	0	0
K	11	12	13	14	15	16	17	18	19	20
ℓ_K	0	0.08	0.15	0.14	0.07	0	0.06	0	0.05	0

5 Conclusions

Time-dependency adds an extra dimension to network science computations, potentially causing a dramatic increase in both storage requirements and computation time. In the case of Katz-style centrality measures, which are based on the solution of linear algebraic systems, allowing for the arrow of time leads naturally to full matrices that keep track of all possible routes for the flow of information. Such a build-up of intermediate data can make large-scale computations unfeasible. In this work, we derived a sparsification technique that delivers accurate approximations to the full-matrix centrality rankings, while retaining the level of sparsity present in the network time-slices. With the new algorithm, as we move forward in time the storage cost remains fixed and the computational cost scales linearly, so the overall task is equivalent to solving a single Katz-style problem at each new time point.

References

1. Acar, E., Dunlavy, D.M., Kolda, T.G.: Link prediction on evolving data using matrix and tensor factorizations. In: ICDMW'09: Proceedings of the 2009 IEEE International Conference on Data Mining Workshops, pp. 262–269 (2009). DOI 10.1109/ICDMW.2009.54
2. Achlioptas, D., Karnin, Z.S., Liberty, E.: Near-optimal entrywise sampling for data matrices. In: C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K.Q. Weinberger (eds.) Advances in Neural Information Processing Systems 26, pp. 1565–1573. Curran Associates, Inc. (2013). URL http://papers.nips.cc/paper/5036-near-optimal-entrywise-sampling-for-data-matrices.pdf
3. Arora, S., Hazan, E., Kale, S.: A fast random sampling algorithm for sparsifying matrices. In: Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, pp. 272–279. Springer (2006)
4. Chen, I., Benzi, M., Chang, H.H., Hertzberg, V.S.: Dynamic communicability and epidemic spread: a case study on an empirical dynamic contact network. Journal of Complex Networks (2016). DOI 10.1093/comnet/cnw017. URL http://comnet.oxfordjournals.org/content/early/2016/06/07/comnet.cnw017.abstract
5. Estrada, E.: The Structure of Complex Networks. Oxford University Press, Oxford (2011)
6. Fagin, R., Kumar, R., Sivakumar, D.: Comparing top k lists. SIAM Journal on Discrete Mathematics **17**(1), 134–160 (2003)
7. Grindrod, P., Parsons, M.C., Higham, D.J., Estrada, E.: Communicability across evolving networks. Physical Review E **83**(4), 046,120 (2011)
8. Holme, P., Saramäki, J.: Temporal networks. Physics Reports **519**, 97–125 (2011)
9. Katz, L.: A new status index derived from sociometric analysis. Psychometrika **18**(1), 39–43 (1953)
10. Laflin, P., Mantzaris, A.V., Grindrod, P., Ainley, F., Otley, A., Higham, D.J.: Discovering and validating influence in a dynamic online social network. Social Network Analysis and Mining **3**, 1311–1323 (2013)
11. Leskovec, J.: SNAP: Network dataset. `https://snap.stanford.edu/data/`
12. Mantzaris, A.V., Higham, D.J.: Asymmetry through time dependency. Eur. Phys. J. B **89**(3), 71 (2016). DOI 10.1140/epjb/e2016-60639-0. URL http://dx.doi.org/10.1140/epjb/e2016-60639-0
13. Mantzaris, A.V., Higham, D.J.: Dynamic communicability predicts infectiousness. In: P. Holme, J. Saramäki (eds.) Temporal Networks, pp. 283–294. Springer, Berlin (2103)
14. Newman, M.E.J.: Networks: An Introduction. Oxford Univerity Press, Oxford (2010)

15. Tang, J., Musolesi, M., Mascolo, C., Latora, V.: Temporal distance metrics for social network analysis. In: Proceedings of the 2nd ACM SIGCOMM Workshop on Online Social Networks (WOSN09). Barcelona (2009)
16. Tang, J., Musolesi, M., Mascolo, C., Latora, V.: Characterising temporal distance and reachability in mobile and online social networks. SIGCOMM Comput. Commun. Rev. **40**, 118–124 (2010)
17. Tang, J., Scellato, S., Musolesi, M., Mascolo, C., Latora, V.: Small-world behavior in time-varying graphs. Physical Review E **81**, 05,510 (2010)
18. Taylor, D., Myers, S.A., Clauset, A., Porter, M.A., Mucha, P.J.: Eigenvector-based centrality measures for temporal networks (2015). ArXiv:1507.01266
19. Wasserman, S., Faust, K.: Social Network Analysis: Methods and Applications. Cambridge University Press, Cambridge (1994)